LEARNED UPLOAD TIME ESTIMATE MODULE

BACKGROUND

[0001] When a user wishes to upload photos to a server via a Yahoo! Photos web site, it is advantageous to display for the user, as a feedback before the upload, an estimate of the amount of time it will take to complete the upload. In order to estimate the upload time for files containing data of the photos before the upload, there are two things needed: 1) total size of the files, and 2) the transfer rate of data.

[0002] There are many factors that can influence the time it take to upload photo image files. These include: time of day, day of week, Internet traffic load, server load, type of Internet connection, number and size of files being uploaded, and the like. Certain times of the day are busier, and certain days of the week are busier. Heavy network traffic load, e.g. Internet or LAN traffic load, can increase upload times as it take longer for data to be serviced, for example, by servers. Likewise, heavy server load increases the delay in responding to upload requests. Also, the type of Internet connection greatly affects the bandwidth of the upload. Besides the transfer rate, the number and size of files to be uploaded affects the upload time. A large number of small files (under 80kb) causes perceived delays in responses from the server, and they cause the underlying internet processing to build up excessively. For each file to upload there is time needed for building up the packets of data, sending the data, and then getting conformation of packet received (success/failure), and each file can have multiple packets to send. Thus, even though a user might have a connection with download speed of 256kb per sec, the upload speed is greatly reduced (sometimes well under 100 kb per sec).

[0003] Current file upload pages have a block of information indicating that it will take certain number of seconds (or minutes), when using a modem, a Digital Subscriber Line (DSL), or other type of connection for uploading a file of a particular size. One way to determine this information is using a process that is stored in the client's 'media' cookie to determine bandwidth. However, not only does this process require user intervention for setting the media cookie, the information obtained detects the 'best case' bandwidth. Namely, although providing a preload time estimate, this approach is a static evaluation based on best case transfer rates rather than a dynamic evaluation based on actual transfer rates. Of course, as files are being uploaded, a special code can determine the actual transfer rate and provide an estimate of remaining time. But, again, this is not a preload estimate. Likewise, hardware tests can be conducted on the user's computer but the results of such tests will not reliably indicate the upload bandwidth, just a 'best' case scenario. Hence there is a need to provide a more realistic indication of the estimated time to complete an upload.

SUMMARY

[0004] Yahoo! has a Photos™ web site where a user can upload their photos to store, share, and order prints. To enhance the effectiveness and user friendliness of the Photos web site, a new downloadable web tool is provided. This web tool includes a new module designed to address the aforementioned need, and it will be hereafter referred to as the "photo uploader" tool.

[0005] This photo uploader tool is downloaded once (when the user initiates the download upon accessing the web page) and its stays resident on the user's computer throughout the entire session. The photo uploader tool is rendered, used, and hosted within the web page. The user can drag-drop photos onto the upload selection control box or use a browse button to select photos. Based on this selection, before the upload, the photo uploader tool provides the upload time estimate to the client (user's computer) for display as feedback to the user. As an example, the display indicates "5 files selected/1.4 MB/65 seconds," where the 1.4 MB is the size of the 5 files, in total, and the time is the total time estimated for the upload of the 5 files. The user then presses the 'upload' button to upload the photos to the Yahoo! Photos™ server

[0006] Advantageously, before each upload, the photo uploader tool establishes a 'learned upload time estimate' which is the actual or 'nearly actual' time it would take to upload data of items currently selected for upload (or items that have been dragged and dropped into the upload selection box). The upload time estimate is determined for each upload in view of historical uploading information which the photo uploader tool gathers, hence the term 'learned upload time estimate.'

[0007] To that end, the photo uploader tool accumulates information on each of the previous uploads (upload size, upload time, timestamp, and number of files), going back to, say, N previous uploads. The photo uploader tool then determines the matching or likeness of the present upload to previous uploads. Namely, the photo loader tool compares information, such as time of day, day of week, and the size, of the file or files to be uploaded against the historical uploading information. A 'match' does not have to be perfect but sufficiently close (i.e. likeness). A 'match' or likeness is determined to exists if the results of the comparison are within a predetermined range or meet a predetermined criteria associated, for example, with traffic load conditions. If a match or likeness is found the actual time (i.e., the total time of the 'matched' previous upload) is used as the time estimate for the upload to be performed. . . If a match or likeness is not found, the average transfer rate for the historical uploads is determined (e.g., in Kbytes per second). The historical average transfer rate is determined by computing the size of the previous uploads, in total, divided by the number of seconds of the previous uploads. Dividing the total size of the current upload by this historical average transfer rate produces the learned upload time estimate (or simply the upload time estimate).

[0008] Certain accommodations are made when there are many small files. Normally, computing the average transfer rate includes dividing the previous upload size of the previous uploads, collectively, by the total time of the previous uploads, collectively, and setting the average transfer rate to the result of this division. However, if the average file size is smaller than this result the average transfer rate equals the average file size per second. If the size of the files is small, the overhead will push the average transfer rate higher closer to the division result, and in that case the average transfer rate is that result.

[0009] In sum, a more realistic estimate is more dependable than a static best case estimate. Thus, as can be appreciated, one advantage of the present invention is that a more realistic estimate of upload time is provided as feedback to the user upon selection of items to upload. This and other features, aspects and advantages of the present invention will become better understood from the description herein, appended claims, and accompanying drawings as hereafter described.

BRIEF DESCRIPTION OF THE DRAWINGS

[00010] The accompanying drawings which, are incorporated in and constitute a part of this specification, illustrate several embodiments of the invention and together with the description, serve to explain the principles of the invention. Wherever convenient, the same reference numbers will be used throughout the drawings to refer to the same or like elements.

FIG. 1A illustrate one system in which the invention is embodied and it shows server-client interactions.

FIG. 1B illustrates the registry settings with historical uploading information.

FIG. 2A is a flow diagram of a client interaction with the photo uploader tool.

FIG. 2B illustrates a Yahoo! Photos ™ web page with the photo uploader feature.

FIG. 2C illustrates a Yahoo! Photos ™ web page with the photo uploader feature once files are selected for uploading.

FIG. 3 is a flow diagram illustrating operations of the learned upload time estimate module.

DETAILED DESCRIPTION

[00011] The present invention is base, in part, on the observation that historical uploading information can facilitate learned upload time estimates. This principle is applied in a new model of a downloadable web tool for establishing upload time estimates. The new model facilitates a more realistic upload time estimate feedback to the user of the Photos web site. The web tool with the new module is designed to address the aforementioned need and it is referred to as the "photo uploader" tool.

[00012] As noted, the salient issue in estimating upload time is the ability to more accurately determine the transfer rate of data. Transfer rate (or baud rate) is the speed with which data can be transmitted between two connected devices; and throughput is the amount of data transferred or processed in a specified amount of time. Data rates are often measured in kilobytes (thousand bytes), megabits (million bits), or megabytes (million bytes) per second. These are usually abbreviated as *KBps, Mbps* and *MBps,* respectively. Thus, connection modems are often referred to by their transfer (or baud) rate, e.g., 33K-modem and 56K-modem. Others are known as LAN (local area network), DSL (digital subscriber line), and Cable modems. Relative to the 56K-modems, LAN, DSL and Cable modems support larger throughputs. In any event, the photo uploader tool is challenged to find the transfer rate in each instance of communications with a user's computer ("client").

[00013] For measuring the actual transfer rate of a connection the most certain way is to upload a file and measure the upload time through that connection. It so happens that the photo uploader tool is first downloaded to the client, and once the photo uploader tool is downloaded it stays resident in the client, unless a new version is available to replace it. Thus, as an ancillary function of the download control, when the photo uploader tool is downloaded a 'test file' can be uploaded to a dummy web page (i.e. to a special server) and the upload time can be measured. However, the question is which file on the client should be selected as the 'test file' to be uploaded. Clearly, it is not a common practice to upload a file without authorization from the user. Moreover, uploading a test file takes time because it requires the test file to be large enough, e.g., 200 Kbyte, to produce a reliable transfer rate. Therefore, although measurement of actual transfer rates optimizes upload time estimates, it is not preferred.

[00014] The preferred approach is one which avoids the forgoing problem. And, although it often results in 'nearly actual' transfer rates and, in turn, 'nearly actual' upload time estimates, rather than always actual estimates, this approach produces more realistic and dependable results compared to conventional 'best-case' estimates. In essence, the learned upload time estimation, which is the preferred approach, tracks file uploads and gathers historical uploading information (e.g., size of previous uploads, number of files, upload time, and upload date/time). This historical information is used to estimate uploads times of subsequent uploads for newly selected files.

[00015] The learned upload time estimation approach is implemented in a system as shown in FIG. 1A. It is noted that the system configuration of FIG. 1A is merely exemplary and other system configurations are suitable without departing from the scope and spirit of the invention. In this configuration, the system 10 includes two servers that the tool interacts with. These servers are conceptually separate and distinct, but in reality they can be one physical unit that performs both functions. The first server is one that 'hosts' the photo uploader tool and it is known as the 'host server' 100. The second server is one to which the tool uploads files (the photo image data files) and it is known as the 'upload server' 102. Although not shown, the client 104 is communicating with the host and upload servers 100 and 102 via a network or networks, i.e., the Internet alone or in any combination with a regional, local or private network.

[00016] The host server 100 contains all the html pages that make up the Photos web site. These web pages allow the user to view photos, share photos with friends, and order reprints of photos. As in other web applications, to access files and albums on the Yahoo! Photos™ web site, the web site requires a unique Yahoo Id and password. The upload server 102 is used for accessing an end-user's files. There are predefined application programming interfaces (APIs) that a client application uses in order to view, upload, and download files from the server. An API is a specific method prescribed by a computer operating system or by an application program by which a programmer writing an application program can make requests of the operating system or another application. Preferably, but not necessarily, the client tool is an ActiveX program written in C++, and it is used within the Microsoft Internet Explorer. It utilizes Microsoft ATL/WTL code libraries to accomplish the various tasks, and it runs on personal computers that use Microsoft Windows® operating system. Note that the photo uploader tool is parameter based, so that it can upload any type of file. It doesn't have to necessarily be an image file, and it can be any file including self-extracting executable (.exe) file. However in the exemplary system the one or more files are JPEG (Joint Photographic Experts Group), GIF (Graphic Interchange Format), PNG (Portable Network Graphics) BMP (bit mapped) formatted files.

[00017] The historical data (i.e., historical uploading information) is stored in a memory associated with the Client 104. For example, the historical data can be maintained in a flat file 112 on the hard drive 106. However, in the present configuration of system 10, the historical data is stored in a hierarchical database known as the 'registry settings' 108. The registry settings are maintained on the hard drive 106 along with the operating system (OS) 110. An example of the registry settings is shown in some detail in FIG. 1B. Registry settings (or simply registry) are stored in a hierarchical database of keys 202 and values 204 representing settings registered in the database in conjunction with the Microsoft Windows® operating system. Typically, the keys and values represent various settings and options necessary to correctly run the operating system, including settings and options for hardware, software, and certain preferences (e.g., minimum disk space, file extensions, timestamp logging, screen saver options, file install/access policies, security configuration and zones, warning levels, etc.). The registry is usually modified by the operating system or an application automatically upon loading the software (e.g., default keys and values). For example, an access key may be registered in the registry upon loading an application. The operating system will use the key in determining whether to grant access to such application. It is also possible to change the registry automatically through application code as is the case with the photo uploader tool. In this instance, a particular area is allocated for the historical uploading information by opening subsections in the registry settings and creating areas for storing this information.

[00018] As shown in FIG. 1B, the historical uploading information is traced back to N previous uploads. The number N can be any number of previous uploads. For example, the historical information can be maintained for 20 previous uploads. The historical information of each upload includes: upload size (size of all files in total) 206, number of files uploaded 208, timestamp (upload start time) 210 and total upload time 212 (e.g., in seconds). For instance, as to upload 9, 'f=3' indicated 3 files, 's=1464.718750' indicates 1464.718750 kilobytes, 't=20' indicates 20 seconds, and 'w=2003-7-29-17-52-2' is the date and time (up to seconds) of upload start ('load' button activated). Note that the timestamp is an upload start event marker that records and represents the upload start event in terms of date and time. However, other upload start event markers are possible without departing from the spirit of the invention. Moreover, the historical uploading information structure can vary from the foregoing and yet the spirit of the invention can be maintained. As will be later explained, these values will be used by the new module in the photo upload tool to establish the learned upload time estimate in a subsequent upload. Then, the history of the subsequent upload will be save for yet another subsequent upload; and the oldest upload information is discarded if the number of previous uploads exceeds N.

[00019] In order to upload photos the user interacts, via the client, with the host and upload servers as illustrated in the flow diagram of FIG. 2A. It is noted that there are many ways of interaction between the photo uploader tool and the Photos web site. To start with, the user navigates its web browser to the Photos web site upload page 302. This provides access to the host server. The Host server supplies an html page that contains a reference (object tag) that causes the photo uploader tool to be downloaded to the client. If the photo uploader tool is not on the client 304, the client will be prompted to download the photo uploader tool 306. If the user refuses the download, the client is linked to another page 308, otherwise the photo uploader tool will be downloaded on the client 310. The html page has parameters set by the host server. These parameters control some of the processing of the photo uploader tool and they include indications as to, for example, which server to upload the files to, which photo album to upload, how many files can the user upload, the maximum upload size (or maximum file size to upload), and the percent variance in file size for matching in transfer rate test. Display literals are also set by the Host.

[00020] Once the photo uploader tool is loaded on the client the image representing it is displayed for the user 316. Note that once the photo uploader tool is loaded, it will call an API on the upload server. This will initialize communications between the photo uploader tool and the upload server. The user is then able to drag and drop photo image files it selects for the current upload 318 onto the photo uploader tool selection feature (or other display feature). Alternatively, the files to be uploaded can be selected one or a few at a time. The photo uploader tool selection feature is shown in FIG. 2B as item 400. Before photos are dragged and dropped, the current selection indication feature 402 shows 0 files selected and the cumulative upload size is 0 bytes. The photos being selected can be previewed in a special window as shown in FIG. 2C. The preview feature is shown as item 404. As can be seen, once photos are selected for uploading, the current selection indication 402 is updated accordingly, and the estimated time to upload the files is provided. This time estimate 406 is the learned upload time estimate.

[00021] Returning to FIG. 2A, the upload time estimate for the current selection is displayed in step 320. If the photo selection is not complete and the user browses for more photos, additional photos are selected for uploading 322. Based on that selection, the photo uploader tool updates the current selection and provides and new upload time estimate 324.

[00022] If the user decides to proceed with the upload, the user selects 'continue/upload' button to prompt the photo uploader tool 326. To that end, the photo uploader tool will locate the files on the client and send the files (one at a time) via an API to the Upload server. As the upload progresses the photo uploader tool provides feedback on the status of the upload 328 and uploads the files to the upload server 330. After uploading all the files the photo uploader tool changes the display to show the number of files successfully uploaded 332, or if failure occurred. Upon completion of uploading all selected files, the photo uploader tool changes the display of the current hosted html page to indicate completion. The user can then go to other portions of the photos web site 334.

[00023] The details of the method for tracking the historical uploading information and establishing the learned upload time estimate are provided in the flow diagram of FIG. 3. After the photo uploader tool is downloaded to the client and displayed to the user, it is ready to accept selection of photo image files. Accordingly, as mentioned above, the user either browses and selects or drags and drops files into the selection feature of the photo uploader tool 502 (as in steps 318 and 322 of FIG. 2A). With the selection of the files for the present upload being complete, the photo uploader tool then turns to its new module to facilitate the learned upload time estimation as well as tracking of the present upload. The new module is referred to as the "learned upload time estimate module."

[00024] Note that tracking of the present upload will produce historical information for use in N subsequent uploads. The number N can vary and in one instance it is set to 20, which means that historical uploading information is traced back to the 20 most recent uploads.

[00025] Therefore, the learned upload time estimate module checks to determine if previous tracking data has been retrieved from the registry settings, file, or any other data structure, designated for holding the historical information for previous uploads 504. If it has not yet been retrieved, the historical information, preferably, for the N most recent uploads is loaded. Note that initially, there will be no historical information available and the current selection indication (item 406 in FIG. 2C) does not include an upload time estimate 510. However, the number will increase from zero to N with each upload that follows. Once the number of uploads increases above N, the historical information for the oldest uploads is discarded leaving historical information only for the most recent N uploads. Also note that other implementations of the invention can choose different methods for deciding how many previous uploads to save, including varying N dynamically.

[00026] Once the learned upload time estimate module determines that one or more previous uploads have been tracked and their historical information is available, it commences a comparison to determine the matching or likeness between information associated with the files currently selected for uploading and the historical uploading information for each of the previous uploads, one previous upload at a time 512.

[00027] In particular, the learned upload time estimate module obtains the aggregate size of the files currently selected for upload, the number of selected files, and the timestamp that indicates the 'upload' selection time (or upload start time). The aggregate size of the selected files is simply termed the "upload size." Then the upload size of the current upload is measured against the size of each of the previous uploads. Note that there need not be a perfect upload size match, and a deviation from the perfect match is allowed within a predefined percentage range. If there is an upload size match, or near match (likeness), with any of previous uploads, the timestamp for the current upload is then compared against the timestamps of previous uploads. Again, there need not be a perfect match, and a near match (likeness) that fits within a predetermined time period or other criteria is possible. The criteria is established to parameterize load conditions, including traffic load conditions experienced by the network. For example, packet traffic load conditions change from day to day, weekday to weekend, and from one time of day to another time of day; although traffic load conditions such as on a Friday night and Saturday evening, or on Friday morning and every other weekday morning, may be similar. Therefore, the learned upload time estimate module is designed with the size and time matching parameters to accommodate various scenarios.

[00028] To illustrate, consider the following scenario. If the timestamp of a previous upload indicates 2003-9-29-15-40-2, and the current time stamp indicates 2003-9-29-21-40-2, it is reasonable to determine that there is no near match because the previous upload occurred during work hours (15:40:02) and the current upload occurs at night (21:40:02). This is true even though both occur on the same day (because at these times the respective traffic load conditions are different). However, if the timestamp of the current upload indicates 2003-9-29-16-40-2, it is a near match because both, the previous upload and the current upload, occur during work hours on the same weekday. In a different scenario, if the timestamp of the previous upload indicates 2003-9-20-19-43-2, and the timestamp of the previous upload indicates 2003-9-21-21-40-2, there is a near match because both, the previous upload and the current upload, occur during the evening hours of the weekend. In yet another example, previous and current uploads that occur on a Friday night and a Saturday evening, respectively, are nearly matched because at these respective times the network experiences similar traffic loads.

[00029] Stated another way, a match exists if any one of the previous upload sizes and the current upload size are similar, and their respective timestamps are similar. Likeness exists if, any one of the previous upload sizes and the current upload size are similar or nearly matched (within a predetermined percent deviation), while their respective timestamps are either similar or fit within a predetermined criteria (parameterized as explained above).

[00030] If based on these parameters a match or likeness is found 512, the actual upload time for the matching previous upload is used as the time estimate for the current upload 514. Namely, the upload time estimate is 'learned' based on the actual upload time of matching or like previous upload and is thus more realistic and dependable than a mere best case estimate.

[00031] However, if based on these parameters a match or likeness is not found with any of the previous uploads, the learned upload time estimate module computes a nearly actual upload time estimate. Again, this computation produces a time estimate that is better than the best case estimate. A salient part of this computation is the computation of the average transfer rate. To that end, the learned upload time estimate module combines the upload size of all previous uploads and, separately or in parallel, it combines the time of all the previous uploads 516. It then divides the total previous uploads size by the total previous uploads time to produce the average transfer rate 518. To compute the learned time estimate, the learned upload time estimate module divides the current upload size (aggregate of selected file sizes) by the average transfer rate 520. The resulting nearly actual upload time estimate is then provided to the client for display to the user 522 (e.g., item 406 in FIG. 2C).

[00032] To illustrate the tracking method during the current upload we turn to the right-side flow diagram of FIG. 3. When the user initiates the upload of the selected files 530, the learned upload time estimate module is prompted to timestamp the start time of the upload 532. As the upload progresses 534 the photo uploader tool monitors the upload and together with the learned upload time estimate module it tracks the upload. One of the monitoring functions is determining if the upload is successful or not, and if a file upload fails an error message is produced. Upon completion of the upload, the learned upload time estimate module determines the time of completion 536 and computes the total upload time 538. It then saves the information for the current upload as historical uploading information for use in subsequent uploads 540. Indeed its timestamp and upload size will be compared against the new timestamp and size of future uploads so as to 'learn' from the current upload the upload time estimate.

[00033] Note that there can be accommodations for small files. That is, if the average file size in the current upload, derived by diving the current upload size by the number of selected files, is smaller than the transfer rate over one second, the smaller average file size can be used for setting the transfer rate in place of the calculated transfer rate.

[00034] In sum, although the present invention has been described in considerable detail with reference to certain preferred versions thereof, other versions are possible. Therefore, the spirit and scope of the appended claims should not be limited to the description of the preferred versions contained herein.